TU Clausthal

Clausthal University of Technology

Model Checking ATL^+ is Harder than It Seemed

Nils Bulling[1] **and Wojciech Jamroga**[2]

IfI Technical Report Series **IfI-09-13**

Department of Informatics
Clausthal University of Technology

Impressum

Publisher: Institut für Informatik, Technische Universität Clausthal
Julius-Albert Str. 4, 38678 Clausthal-Zellerfeld, Germany
Editor of the series: Jürgen Dix
Technical editor: Michael Köster
Contact: michael.koester@tu-clausthal.de
URL: http://www.in.tu-clausthal.de/forschung/technical-reports/
ISSN: 1860-8477

The IfI Review Board

Prof. Dr. Jürgen Dix (Theoretical Computer Science/Computational Intelligence)
Prof. i.R. Dr. Klaus Ecker (Applied Computer Science)
Prof. Dr. Barbara Hammer (Theoretical Foundations of Computer Science)
Prof. Dr. Sven Hartmann (Databases and Information Systems)
Prof. Dr. Kai Hormann (Computer Graphics)
Prof. i.R. Dr. Gerhard R. Joubert (Practical Computer Science)
apl. Prof. Dr. Günter Kemnitz (Hardware and Robotics)
Prof. i.R. Dr. Ingbert Kupka (Theoretical Computer Science)
Prof. i.R. Dr. Wilfried Lex (Mathematical Foundations of Computer Science)
Prof. Dr. Jörg Müller (Business Information Technology)
Prof. Dr. Niels Pinkwart (Business Information Technology)
Prof. Dr. Andreas Rausch (Software Systems Engineering)
apl. Prof. Dr. Matthias Reuter (Modeling and Simulation)
Prof. Dr. Harald Richter (Technical Computer Science)
Prof. Dr. Gabriel Zachmann (Computer Graphics)
Prof. Dr. Christian Siemers (Hardware and Robotics)

Model Checking $\mathbf{ATL}^+$ is Harder than It Seemed

Nils Bulling[1] and Wojciech Jamroga[2]

[1] Department of Informatics, Clausthal University of Technology, Germany
[2]Computer Science and Communications, University of Luxembourg
bulling@in.tu-clausthal.de, wojtek.jamroga@uni.lu

Abstract

$\mathbf{ATL}^+$ is a variant of the alternating-time temporal logic that does not have the expressive power of full $\mathbf{ATL}^*$, but still allows for expressing some natural properties of agents. It has been believed that verification with $\mathbf{ATL}^+$ is $\boldsymbol{\Delta}_3^{\mathbf{P}}$-complete for both memoryless agents and players who can memorize the whole history of the game. In this paper, we show that the latter result was not correct. That is, we prove that model checking $\mathbf{ATL}^+$ for agents that use strategies with memory is in fact **PSPACE**-complete. On a more optimistic note, we show that fairness constraints can be added to $\mathbf{ATL}^+$ without further increasing the complexity of model checking, which makes $\mathbf{ATL}^+$ an attractive alternative to the full language of $\mathbf{ATL}^*$.

1 Introduction

The alternating-time temporal logic $\mathbf{ATL}^*$ and its less expressive version **ATL** [1, 3] have been studied extensively in previous years. Much research was focused on the way such logics can be used for verification of multi-agent systems, *model checking* being the most important method in this respect. Consequently, the computational complexity of model checking turned out essential to evaluate and compare the practical usability of different variants of strategic logics.

It is known that model checking problem of full $\mathbf{ATL}^*$ is doubly exponential time-complete, and only polynomial time-complete for **ATL**, if perfect recall strategies are used [3], i.e., if players can memorize the whole history of the game. Hence, the latter logic is more attractive computationally. However, there is also a price to pay in terms of expressiveness. The simple property that an agent can make p true infinitely often can, for instance, be expressed in $\mathbf{ATL}^*$ but not in **ATL**.

A tradeoff is offered by $\mathbf{ATL}^+$, a variant of the alternating-time temporal logic that does not have the expressive power of full $\mathbf{ATL}^*$, but still allows

for expressing some natural properties of agents. For example, we can use formula $\langle\langle robot \rangle\rangle(\Diamond \mathsf{cleanRoom} \wedge \Diamond \mathsf{packageDelivered})$ to demand that the robot can clean the room and deliver the package, without specifying in which order the tasks will be accomplished.

Verification with $\mathbf{ATL}^+$ is believed to be $\mathbf{\Delta_3^P}$-complete for both memoryless and perfect recall strategies [14]. In this paper, we show that the latter result is wrong. That is, we prove that model checking $\mathbf{ATL}^+$ for agents that use strategies with full memory is in fact $\mathbf{PSPACE}$-complete. Since the $\mathbf{\Delta_3^P}$-completeness for the memoryless semantics still holds, we get that memory makes verification harder already for $\mathbf{ATL}^+$, and not just for $\mathbf{ATL}^*$ as it was believed before. We also show that fairness conditions can be added to $\mathbf{ATL}^+$ without further increasing the complexity.

The rest of this paper is structured as follows. In Section 2 we introduce the relevant logics and their models, and discuss the variant $\mathbf{ATL}^+$ in more detail. In Section 3 we correct the existing "results" on model checking complexity of $\mathbf{ATL}^+$ for agents with perfect information and perfect recall. In Section 4 we study $\mathbf{EATL}^+$, i.e., $\mathbf{ATL}^+$ augmented with the temporal operator $\Box\Diamond$ ("infinitely often"). Finally, we present some conclusions in Section 5.

2 ATL$^+$ and the Matter of Recall

We begin by introducing the strategic logics that will be discussed in this paper. The *alternating-time temporal logic* $\mathbf{ATL}^*$ [1, 3] is a temporal logic that incorporates some basic game-theoretical notions. Essentially, $\mathbf{ATL}^*$ generalizes the branching time logic $\mathbf{CTL}^*$ [5, 6] by replacing path quantifiers E, A with so called *cooperation modalities* $\langle\langle A \rangle\rangle$. Informally, $\langle\langle A \rangle\rangle\gamma$ expresses that the group of agents A has a collective strategy to enforce temporal property γ. $\mathbf{ATL}^*$ formulae include temporal operators: "$\bigcirc$" ("in the next state") and $\mathcal{U}$ ("until"). Additional operators "$\Diamond$" ("now or sometime in the future") and "$\Box$" ("always from now on") can be defined as $\Diamond\gamma \equiv \top\,\mathcal{U}\,\gamma$ and $\Box\gamma \equiv \neg\Diamond\neg\gamma$. It should be noted that the path quantifiers A, E of $\mathbf{CTL}^*$ can be expressed in $\mathbf{ATL}^*$ with $\langle\langle\emptyset\rangle\rangle, \langle\langle\mathbb{A}\mathrm{gt}\rangle\rangle$ respectively.

2.1 Syntax and Variants

In the rest of the paper we assume that Π is a nonempty set of *proposition symbols* and $\mathbb{A}\mathrm{gt}$ a nonempty and finite set of *agents*. Alternating-time temporal logic comes in several variants, of whom $\mathbf{ATL}^*$ is the broadest. Formally, the language of $\mathbf{ATL}^*$ is given by formulae φ generated by the grammar below, where $A \subseteq \mathbb{A}\mathrm{gt}$ is a set of agents, and $\mathsf{p} \in \Pi$ is an atomic proposition:

$$\varphi ::= \mathsf{p} \mid \neg\varphi \mid \varphi \wedge \varphi \mid \langle\langle A \rangle\rangle\gamma,$$

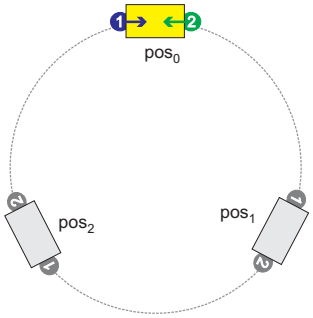

Figure 1: Two robots and a carriage: a schematic view

$$\gamma ::= \varphi \mid \neg\gamma \mid \gamma \wedge \gamma \mid \bigcirc\gamma \mid \gamma \mathcal{U} \gamma.$$

Formulae φ are called *state formulae*, and γ *path formulae* of **ATL***.

The best known variant of the alternating time temporal logics is **ATL** ("**ATL** without star" or "vanilla" **ATL**) in which every occurrence of a cooperation modality is immediately preceded by exactly one temporal operator. In this paper, however, we study the model checking problem for **ATL**$^+$, a variant that sits between **ATL*** and **ATL**. The language of **ATL**$^+$ includes only formulae where each temporal operator is followed by a state formula, which allows cooperation modalities to be followed by *a Boolean combination* of path subformulae. Formally, **ATL**$^+$ formulae are defined by the following grammar:

$$\varphi ::= \mathsf{p} \mid \neg\varphi \mid \varphi \wedge \varphi \mid \langle\langle A \rangle\rangle \gamma,$$
$$\gamma ::= \neg\gamma \mid \gamma \wedge \gamma \mid \bigcirc\varphi \mid \varphi \mathcal{U} \varphi.$$

Example 1 *The* **ATL** *formula* $\langle\langle jamesbond \rangle\rangle \Diamond \mathsf{win}$ *says that James Bond can eventually win, no matter how the other agents act. On the other hand,* $\langle\langle jamesbond \rangle\rangle \Box(\mathsf{missionAssigned} \rightarrow \Diamond \mathsf{MissionAccomplished})$ *is an* **ATL*** *formula (which clearly belongs to neither* **ATL** *nor* **ATL**$^+$*) that deems agent 007 able to accomplish all his future missions. Finally,* $\langle\langle jamesbond \rangle\rangle (\Box\neg\mathsf{crash} \wedge \Diamond \mathsf{land})$ *(James Bond can prevent the space ship from crashing and make it eventually land) is a formula of* **ATL**$^+$ *but not of* **ATL**.

2.2 Semantics

The semantics of **ATL*** is defined over a variant of transition systems where transitions are labeled with combinations of actions, one per agent. Formally, a *concurrent game structure* (**CGS**) is a tuple $M = \langle \mathbb{A}\mathrm{gt}, St, \Pi, \pi, Act, d, o \rangle$

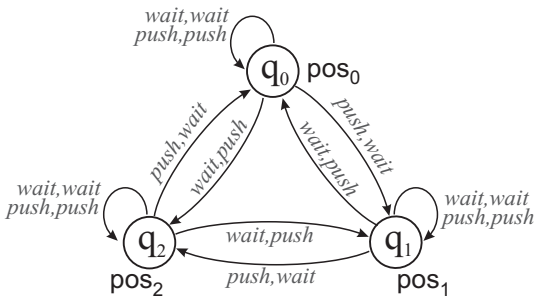

Figure 2: Two robots and a carriage: concurrent game structure M_1 that models the scenario

which includes a nonempty finite set of all agents $\mathbb{A}\mathrm{gt} = \{1, \ldots, k\}$, a nonempty set of states St, a set of atomic propositions Π and their valuation $\pi : \Pi \to 2^{St}$, and a nonempty finite set of (atomic) actions Act. Function $d : \mathbb{A}\mathrm{gt} \times St \to 2^{Act}$ defines nonempty sets of actions available to agents at each state, and o is a (deterministic) transition function that assigns the outcome state $q' = o(q, \alpha_1, \ldots, \alpha_k)$ to state q and a tuple of actions $\langle \alpha_1, \ldots, \alpha_k \rangle$ for $\alpha_i \in d(i, q)$ and $1 \leq i \leq k$, that can be executed by $\mathbb{A}\mathrm{gt}$ in q. Thus, we assume that all the agents execute their actions synchronously; the combination of the actions, together with the current state, determines the next transition of the system.

In the rest of the paper, we will write $d_i(q)$ instead of $d(i, q)$, and we will denote the set of collective choice of group A at state q by $d_A(q) = \prod_{i \in A} d_i(q)$. A *path* $\lambda = q_0 q_1 q_2 \ldots$ is an infinite sequence of states such that there is a transition between each q_i, q_{i+1}. We use $\lambda[i]$ to denote the ith position on path λ (starting from $i = 0$) and $\lambda[i, \infty]$ to denote the subpath of λ starting from i.

Example 2 (Robots and Carriage) *Consider the scenario depicted in Figures 1 and 2. Two robots push a carriage from opposite sides. As a result, the carriage can move clockwise or anticlockwise, or it can remain in the same place. We assume that each robot can either push (action $push$) or refrain from pushing (action $wait$). Moreover, they both use the same force when pushing. Thus, if the robots push simultaneously or wait simultaneously, the carriage does not move. When only one of the robots is pushing, the carriage moves accordingly.*

To make our model of the domain discrete, we identify 3 different positions of the carriage, and associate them with states q_0, q_1, and q_2. We label the states with propositions $\mathsf{pos}_0, \mathsf{pos}_1, \mathsf{pos}_2$, respectively, to allow for referring to the current position of the carriage in the object language.

A *strategy* of agent a is a plan that specifies what a is going to do in each situation. It makes sense, from a conceptual and computational point of view, to distinguish between two types of strategies: an agent may base his decision on the current state or on the whole history of events that have hap-

pened. Also, the agent may have complete or incomplete knowledge about the current global state of the system throughout the game. Hence, there are four classes of strategies that are obtained by mixing *imperfect (i)* (resp. *perfect (I)*) *information* and *imperfect (r)* (resp. *perfect (R)*) *recall* (cf. [14] for a discussion of these variants). Here we are mainly interested in the perfect information/prefect recall setting.

A *perfect information perfect recall strategy* (called *IR-strategy* in the rest of the paper) for agent a is a function $s_a : St^+ \to Act$ such that $s_a(q_0q_1 \dots q_n) \in d_a(q_n)$. A *memoryless strategy* (called *Ir-strategy* in the rest of the paper), on the other hand, is a function $s_a : St \to Act$ such that $s_a(q) \in d_a(q)$. Alternatively, memoryless strategies can be seen as perfect recall strategies such that $s_a(hq) = s_a(h'q)$ for all $h, h' \in St^+$.

A *collective strategy* for a group of agents $A = \{a_1, \dots, a_r\}$ is simply a tuple of individual strategies $s_A = \langle s_{a_1}, \dots, s_{a_r} \rangle$. By $s_A|_a$, we denote agent a's part s_a of the collective strategy s_A where $a \in A$. Function $out(q, s_A)$ returns the set of all paths that may occur when agents A execute strategy s_A from state q onward:

$out(q, s_A) = \{\lambda = q_0q_1q_2 \dots \mid q_0 = q$ and for each $i = 1, 2, \dots$ there exists a tuple of agents' decisions $\langle \alpha^{i-1}_{a_1}, \dots, \alpha^{i-1}_{a_k} \rangle$ such that $\alpha^{i-1}_a \in d_a(q_{i-1})$ for every $a \in \mathbb{A}\mathrm{gt}$, and $\alpha^{i-1}_a = s_A|_a(q_0q_1 \dots q_{i-1})$ for every $a \in A$, and $o(q_{i-1}, \alpha^{i-1}_{a_1}, \dots, \alpha^{i-1}_{a_k}) = q_i\}$.

Let M be a **CGS**, and q a state in M. The (perfect recall) *IR*-semantics of $\mathbf{ATL}^*$ is defined by the following clauses [3, 14]:

$M, q \models \mathsf{p}$ iff $\lambda[0] \in \pi(\mathsf{p})$ and $\mathsf{p} \in \Pi$;

$M, q \models \neg\varphi$ iff $M, q \not\models \varphi$;

$M, q \models \varphi \wedge \psi$ iff $M, q \models \varphi$ and $M, q \models \psi$;

$M, q \models \langle\langle A \rangle\rangle\gamma$ iff there is a strategy s_A for agents A such that for each path $\lambda \in out(s_A, q)$, we have $M, \lambda \models \gamma$.

$M, \lambda \models \varphi$ iff $M, \lambda[0] \models \varphi$;

$M, \lambda \models \neg\gamma$ iff $M, \lambda \not\models \gamma$;

$M, \lambda \models \gamma \wedge \delta$ iff $M, \lambda \models \gamma$ and $M, \lambda \models \delta$;

$M, \lambda \models \bigcirc\gamma$ iff $\lambda[1, \infty], \pi \models \gamma$; and

$M, \lambda \models \gamma\,\mathcal{U}\,\delta$ iff there is an $i \in \mathbb{N}_0$ such that $M, \lambda[i, \infty] \models \delta$ and $M, \lambda[j, \infty] \models \gamma$ for all $0 \leq j < i$;

The *(memoryless) Ir-semantics* is defined analogously, with memoryless strategies used instead. Accordingly, we use $\mathbf{ATL}^*_{\mathbf{IR}}$, $\mathbf{ATL}_{\mathbf{Ir}}$, $\mathbf{ATL}^+_{\mathbf{IR}}$, $\mathbf{ATL}^+_{\mathbf{Ir}}$, $\mathbf{ATL}_{\mathbf{IR}}$, and $\mathbf{ATL}_{\mathbf{Ir}}$ to refer to the language using the semantics the names are annotated with. Note that teh *IR*-semantics are considered as the canonical semantics if nothing is stated.

Note that the same semantics can be used for sublanguages of **ATL*** (here: **ATL**$^+$ and **ATL**). It is worth mentioning that a completely state-based semantics can be given for **ATL**, which underlies the polynomial-time model checking algorithm [3].

Example 3 (Robots and Carriage, ctd.) *Since the outcome of each robot's action depends on the current action of the other robot, no agent can make sure that the carriage moves to any particular position. So, we have for example that* $M_1, q_0 \models \neg\langle\langle 1\rangle\rangle\Diamond \mathsf{pos}_1$. *On the other hand, the robots can cooperate to move the carriage. For instance, it holds that* $M_1, q_0 \models \langle\langle 1,2\rangle\rangle\Diamond \mathsf{pos}_1$ *(example strategy: robot* 1 *always pushes and robot* 2 *always waits).*

In fact, the same strategy can be used to express that the robots can make the carriage visit every *position, which is captured by the following* **ATL**$^+$ *satisfaction:* $M_1, q_0 \models \langle\langle 1,2\rangle\rangle(\Diamond \mathsf{pos}_0 \wedge \Diamond \mathsf{pos}_1 \wedge \Diamond \mathsf{pos}_2)$. *Note that all the above properties hold regardless of whether we talk about memoryless agents or agents with perfect recall.*

2.3 Some Known Results. Importance of ATL$^+$

It is well known that $\mathbf{ATL_{Ir}}$ =$\mathbf{ATL_{IR}}$ [3, 14]. That is, if agents A have a perfect recall strategy to bring about φ they also have a memoryless strategy to bring about it. The same is *not* true for **ATL***, and in fact, already for **ATL**$^+$. For example, formula $\langle\langle 1,2\rangle\rangle(\Diamond \mathsf{pos}_1 \wedge \Diamond \mathsf{pos}_2)$ holds in M_1, q_0 in the set of perfect recall strategies but not in the set of memoryless strategies. In consequence, **ATL**$^+$ can be seen as a minimal well-known variant of the alternating-time logic that discerns the memoryless and perfect recall semantics.

In conceptual terms, **ATL**$^+$ allows for reasoning about what can be achieved under certain assumptions about the agents' behavior. This kind of properties has been especially studied in deontic logic and normative systems (e.g., [11, 12, 16]), but also in reasoning about plausible behavior of agents [4]. For instance, consider a class of systems where a state is labeled with proposition V_a iff agent a has violated a social norm with his last action. Then, property "coalition A can enforce property γ under the assumption that everybody adheres to social norms" can be expressed as an **ATL**$^+$ formula $\langle\langle A\rangle\rangle\big(\bigwedge_{a\in A}\Box\mathsf{V}_a \wedge (\bigwedge_{a\notin A}\Box\mathsf{V}_a \rightarrow \gamma)\big)$. Moreover, we can use **ATL**$^+$ to specify a set of goals that should be achieved without saying in which order they must be accomplished, e.g., in formula $\langle\langle robot\rangle\rangle(\Diamond\mathsf{cleanRoom} \wedge \Diamond\mathsf{packageDelivered})$.

Contrary to popular belief, **ATL**$^+$ *is* more expressive than **ATL**, which follows from the fact that the "weak until" operator is expressible in **ATL**$^+$ (as $\varphi\,\mathcal{W}\,\psi \equiv \varphi\,\mathcal{U}\,\psi \vee \Box(\neg\psi)$), but not in the original version of **ATL** [9]. Still, some formulae of **ATL**$^+$ have their equivalent counterparts in **ATL**. For instance, $\langle\langle jamesbond\rangle\rangle(\Box\neg\mathsf{crash} \wedge \Diamond\mathsf{land})$ from Example 1 can be equivalently rephrased as $\langle\langle jamesbond\rangle\rangle(\neg\mathsf{crash})\,\mathcal{U}\,(\mathsf{land} \wedge \langle\langle jamesbond\rangle\rangle\Box\neg\mathsf{crash})$.

In particular, in the perfect recall semantics, we have that **ATL**$^+$ formulae can be equivalently translated into **ATL** with the "weak until" opera-

tor [7]. We observe that in some cases the translation results in an exponential blowup of the length of the formula. Thus, $\mathbf{ATL}^+$ has the same expressive power as $\mathbf{ATL}$ with "weak until" over the semantics used here – but it allows for exponentially more succinct and intuitive specifications of some properties (this follows from the results in [15]).

Regarding the complexity of verification for strategic logics, the following patterns can be observed (albeit not without exceptions):

- Model checking more expressive and/or succinct logics is usually harder than the less expressive/succinct ones;
- Model checking imperfect information agents is usually harder than perfect information ones;
- Model checking agents with perfect recall is usually harder than memoryless agents.

Indeed, model checking of $\mathbf{ATL_{Ir}}$ can be done in linear time wrt the number of transitions in the model and the length of the formula [3],[1] while model checking of $\mathbf{ATL^+_{Ir}}$ is $\mathbf{\Delta_3^P}$-complete,[2] and model checking of $\mathbf{ATL^*_{Ir}}$ is $\mathbf{PSPACE}$-complete. Moreover, model checking of $\mathbf{ATL_{IR}}$ is still linear (it is *the same* logic after all) while verification of $\mathbf{ATL^*_{IR}}$ is complete in double exponential time [3].

What about model checking $\mathbf{ATL^+_{IR}}$? In [14], it is claimed to be $\mathbf{\Delta_3^P}$-complete, so apparently no price is paid for assuming agents' memory in this case. Unfortunately, the claim is wrong. We will show in Section 3 that the problem becomes $\mathbf{PSPACE}$-complete in the perfect recall setting. Note that the results in [9] that concern the complexity of $\mathbf{ATL^+_{IR}}$ in various non-standard settings are also incorrect, since they crucially depend on the claim from [14]; we will correct them in Section 3.3.

3 Model Checking ATL$^+$ with Recall Is PSPACE-Complete

In an excellent study [14], Schobbens claims that model checking $\mathbf{ATL}^+$ is $\mathbf{\Delta_3^P}$-complete wrt to the number of transitions in the model and the length of the formula, for both perfect recall and memoryless semantics. For memoryless agents, the upper bound can be shown by the following algorithm. Given a formula $\langle\langle A\rangle\rangle\gamma$ with no nested cooperation modalities, we can guess

[1] It is important to add that that the "weak until" operator $\mathcal{W}$ does not increase the complexity [9].

[2] $\mathbf{\Delta_3^P} = \mathbf{P}^{\mathbf{\Sigma_2^P}}$ is the class of problems that can be solved by a deterministic Turing machine that can make adaptive queries to an oracle of type $\mathbf{\Sigma_2^P} = \mathbf{NP}^{\mathbf{NP}}$. That is, the oracle is a nondeterministic TM that can query another oracle (a nondeterministic TM itself). All the three machines are required to run in polynomial time.

a memoryless strategy of A, "trim" the model accordingly, and model-check the $\mathbf{CTL}^+$ formula $\mathsf{A}\gamma$ in the resulting model. Note that a memoryless strategy can be guessed in polynomially many steps, and the trimming process requires only polynomially many steps too. For nested cooperation modalities, we repeat the procedure recursively (bottom-up). Since model checking of the $\mathbf{CTL}^+$ formula $\mathsf{A}\gamma$ can be done in nondeterministic polynomial time, we get that the overall procedure runs in time $\mathbf{\Delta_2^{P^{NP}}} = \mathbf{\Delta_3^P}$ [14].

For agents with perfect recall, a similar argument *seems* correct. Every formula of $\mathbf{ATL}^+$ can be translated to an equivalent formula of $\mathbf{ATL}$ with weak until [7], and for $\mathbf{ATL}$ (also with weak until) it does not make a difference whether the perfect recall or memoryless semantics is used, so memoryless strategies can be used instead of memoryful ones. Hence, it is enough to guess a memoryless strategy, trim the model etc. Unfortunately, this line of reasoning is wrong because the result of the translation (the $\mathbf{ATL}$ formula) may include *exponentially many* cooperation modalities (instead of one in the original $\mathbf{ATL}^+$ formula). For example, formula $\langle\!\langle A\rangle\!\rangle(\Diamond\mathsf{p} \wedge \Diamond\mathsf{q})$ is translated to $\langle\!\langle A\rangle\!\rangle\Diamond\big((\mathsf{p} \wedge \langle\!\langle A\rangle\!\rangle\Diamond\mathsf{q}) \vee (\mathsf{q} \wedge \langle\!\langle A\rangle\!\rangle\Diamond\mathsf{p})\big)$; for a longer list of achievement goals ($\Diamond\mathsf{p}_i$) every permutation must be explicitly enumerated. Thus, we may need to guess exponentially many polynomial-size strategies, which clearly cannot be done in polynomial time.

There seems to be an intuitive way of recovering from the problem. Note that, in an actual execution, only a polynomial number of these strategies will be used. So, we can try to first guess a sequence of goals (in the right order) for whom strategies will be needed, then the strategies themselves, fix those strategies in the model (cloning the model into as many copies as we need) and check the corresponding $\mathbf{CTL}^+$ formula in it. Unfortunately, this is also wrong: for different execution paths, we may need *different* ordering of the goals (and hence strategies). And we have to consider exponentially many paths in the worst case.

So, what is the complexity of model checking $\mathbf{ATL}^+_{\mathbf{IR}}$ at the end? The problem turns out to be much harder than $\mathbf{\Delta_3^P}$, namely $\mathbf{PSPACE}$-complete.

3.1 Lower Bound

We prove the $\mathbf{PSPACE}$-hardness by a reduction of Quantified Boolean Satisfiability (QSAT), a canonical $\mathbf{PSPACE}$-complete problem.

Definition 1 (QSAT [13]) ***Input:*** *A Boolean formula Φ in negation normal form (i.e., negation occurs only in literals) with n propositional variables $x_1, \ldots, x_n$.* ***Output:*** *True if $\exists x_1 \forall x_2 \ldots Q_n x_n\ \Phi$ holds, false otherwise (where $Q_n = \forall$ if n is even, and $Q_n = \exists$ if n is odd).*

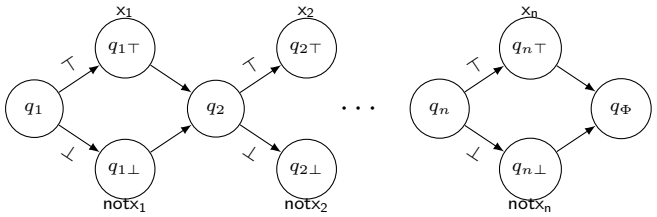

Figure 3: Construction of the concurrent game structure for QSAT: value choice section

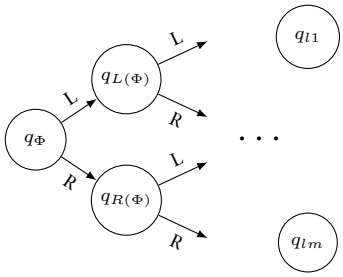

Figure 4: **CGS** for QSAT: formula structure section

Given an instance of QSAT we construct a turn-based[3] concurrent game structure M with two players: the *verifier* $\mathbf{v}$ and the *refuter* $\mathbf{r}$. The structure consists of the following sections:

- *Value choice section:* a sequence of states q_i, one per variable x_i, where the values of x_i's will be "declared", see Figure 3. States q_i with odd i are controlled by $\mathbf{v}$, states with even i are controlled by $\mathbf{r}$. The owner of a state can choose between two possible valuations $(\top, \bot)$. Choosing $\top$ leads to a state where the proposition $\mathsf{x_i}$ holds; choosing $\bot$ leads to a state labeled by the proposition $\mathsf{notx_i}$.
- *Formula structure section:* corresponds to the parse tree of Φ, see Figure 4. For every subformula Ψ of Φ, there is a state q_Ψ with two choices: L leading to state $q_{L(\Psi)}$ and R leading to $q_{R(\Psi)}$, where $L(\Psi)$ is the left hand side subformula of Ψ and $R(\Psi)$ is the right hand side subformula of Ψ. The verifier controls q_Ψ if the outermost connective in Ψ is a disjunction; the refuter controls the state if it is a conjunction.
- *Sections of literals:* for every literal l in Φ, a single state q_l is created, controlled by the owner of the Boolean variable x_i in l. Like in the value

[3] A model is *turn-based* if each state has a single agent that controls the subsequent transition, and the other agents have no real choice there (which can be modeled by assuming $d_q(a) = \{wait\}$ for every agent a except the "owner" of q).

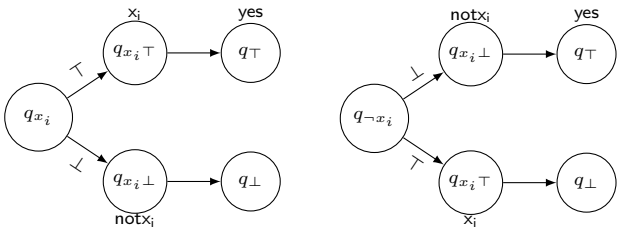

Figure 5: **CGS** for QSAT: sections of literals

choice section, the agent chooses a value ($\top$ or $\bot$) for the variable (not for the literal!) which leads to a new state labeled with the proposition $\mathsf{x_i}$ (for action $\top$) or $\mathsf{notx_i}$ (for $\bot$). Finally, the system proceeds to the winning state $q_\top$ (labeled with the proposition yes) if the valuation of x_i made the literal l true, and to the losing state $q_\bot$ otherwise – see Figure 5 for details.

Let us define an auxiliary "consistency" macro: $\mathsf{Cons}_i \equiv \Box\neg\mathsf{x_i} \vee \Box\neg\mathsf{notx_i}$ which expresses that the value of x_i cannot be declared both $\top$ and $\bot$ during a single execution of the model. Now, we have the following:

Lemma 1 $\exists x_1 \forall x_2 \ldots Q_n x_n\, \Phi$ *iff*

$$M, q_1 \models_{IR} \langle\langle \mathbf{v} \rangle\rangle \Big(\bigwedge_{i \in Odd} \mathsf{Cons}_i \wedge \big(\bigwedge_{i \in Even} Cons_i \rightarrow \Diamond \mathsf{yes} \big) \Big).$$

Proof sketch The **ATL**$^+$ formula specifies that **v** can consistently assign values to "his" variables, so that if **r** consistently assigns values to "his" variables (in any way), formula Φ will always evaluate to $\top$, which is exactly the meaning of QSAT. The way a player assigns a value to variable x_i may depend on what has been assigned to $x_1, \ldots, x_{i-1}$. Note that this is the reason why perfect recall is necessary to obtain the reduction. ■

We observe that the construction results in a model with $O(|\Phi|)$ states and transitions, and it can be constructed in $O(|\Phi|)$ steps, so we get the following result.

Theorem 2 *Model checking* $\mathbf{ATL^+_{IR}}$ *is* **PSPACE***-hard with respect to the number of transitions in the model and the length of the formula. It is* **PSPACE***-hard even for turn-based models with two agents and "flat"* $\mathbf{ATL^+_{IR}}$ *formulae, i.e., ones that include no nested cooperation modalities.*

3.2 Upper Bound

In this section we show that model checking $\mathbf{ATL}^+_{\mathbf{IR}}$ can be done in polynomial space. Our proof has been inspired by the construction in [10], proposed for $\mathbf{CTL}^+$. We begin by introducing some notation.

Definition 2 (Strategy for (M, q, γ)) *We say that $s_A \in \Sigma_A$ is a* strategy for (M, q, γ) *if for all $\lambda \in out_M(q, s_A)$ it holds that $M, \lambda \models \gamma$.*

An $\mathbf{ATL}^+$-path formula γ is called *atomic* if it has the form $\bigcirc \varphi_1$ or $\varphi_1 \mathcal{U} \varphi_2$ where $\varphi_1, \varphi_2 \in \mathbf{ATL}^+$. For $\varphi \in \mathbf{ATL}^+$ we denote the set of all atomic path subformulae of φ by $\mathcal{APF}(\varphi)$. And, as before, we call an $\mathbf{ATL}^+$-path formula γ *flat* if it does not contain any more cooperation modalities.

Now we can define the notion of *witness position*, that is, a specific position on a path that "makes" a path formula true or false.

Definition 3 (Witness position) *Let γ be a flat atomic path formula, and let λ be a path. The* witness position $witpos_M(\lambda, \gamma)$ *of γ wrt λ is defined as follows:*
(1) if $\gamma = \bigcirc \varphi$ then $witpos_M(\lambda, \gamma) = 1$;
(2) if $\gamma = \varphi_1 \mathcal{U} \varphi_2$ and

- *$\lambda \models \gamma$ then $witpos_M(\lambda, \gamma) = \min\{i \geq 0 \mid \lambda[i] \models \varphi_2\}$*
- *$\lambda \not\models \gamma$ and $\lambda \models \Diamond \varphi_2$ then $witpos_M(\lambda, \gamma) = \min\{i \geq 0 \mid \lambda[i] \models \neg(\varphi_1 \wedge \varphi_2)\}$*
- *$\lambda \not\models \gamma$ and $\lambda \not\models \Diamond \varphi_2$ then $witpos_M(\lambda, \gamma) = -1$.*

Moreover, for a flat (not necessarily atomic) $\mathbf{ATL}^+$ path formula γ, we define the set of witness positions of γ wrt λ as $wit_M(\lambda, \gamma) = (\bigcup_{\gamma' \in \mathcal{APF}(\gamma)} \{witpos_M(\lambda, \gamma')\}) \cap \mathbb{N}_0$.

For instance, if formula $\Box\neg\mathsf{p}$ is true on λ then $witpos_M(\lambda, \Box\neg\mathsf{p}) = -1$ since the formula is an abbreviation for $\neg(\top \mathcal{U} \mathsf{p})$, and for this formula we have that $w = -1$ on λ. In consequence, $wit_M(\lambda, \Box\neg\mathsf{p}) = \emptyset$.

In the next lemma we show that if there is a strategy that enforces a (flat) path formula γ then the witnesses of all atomic subformulae of γ can be found in a bounded initial fragment of each resulting path. Firstly, we introduce the notion of a *segment* which can be seen as a "minimal loop".

Definition 4 (Segment) *A* segment *of path λ is a tuple $(i, j) \in \mathbb{N}_0^2$ with $i < j$ such that $\lambda[i] = \lambda[j]$ and there are no index k, k' with $i < k < k' < j$ such that $\lambda[k] = \lambda[i]$ or $\lambda[k] = \lambda[k']$. The set of segments of λ is denoted by $seg(\lambda)$.*

Lemma 3 *Let $M, q \models \langle\langle A \rangle\rangle \gamma$. Then, there is a strategy s_A for (M, q, γ) such that for all paths $\lambda \in out_M(q, s_A)$ the following property holds: For every segment $(i, j) \in seg(\lambda)$ with $j \leq \max wit_M(\lambda, \gamma)$ there is a witness position $k \in wit_M(\lambda, \gamma)$ with $i \leq k \leq j$.*

Proof Suppose such a strategy does not exist; then, for any strategy s_A for (M, q, γ), there is a path $\lambda \in out(q, s_A)$ and a segment $(i, j) \in seg(\lambda)$ with $j \leq \max wit_M(\lambda, \gamma)$ such that there is no $k \in wit_M(\lambda, \gamma)$ with $i \leq k \leq j$.

We now define s'_A as the strategy that is equal to s_A except that it cuts out the "idle" segment (i, j) from λ, i.e., $s'_A(\lambda[0, i]h) := s_A(\lambda[0, j]h)$ for all $h \in St^+$, and $s'_A(h) := s_A(h)$ otherwise.

Let $[h]_{q,s_A}$ denote the set of all paths λ' such that $h\lambda' \in out(q, s_A)$ where $h \in St^+$. Now it is easy to see that for all $\lambda' \in [\lambda[0, j]]_{q,s_A}$ we have that the path $\lambda[0, j]\lambda'$ satisfies γ if, and only if, the path $\lambda[0, i]\lambda'$ does since either both paths belong to the outcome or neither of them. Hence, we have that all paths in $out(q, s'_A)$ satisfy γ. Moreover, the latter set of outcomes is non-empty iff $out(q, s_A)$ is not which proves the Lemma. By following this procedure recursively, we obtain a strategy that reaches a witness in every segment of λ up to $\max wit_M(\lambda, \gamma)$. ∎

Given, for instance, an **ATL**$^+$ formula $\langle\langle A\rangle\rangle(\Diamond \mathsf{p} \wedge \Diamond \mathsf{r})$ the previous lemma says that if A have any winning strategy than they also have one such that only the first two segments on each path in the outcome are important to witness the truth of $\Diamond \mathsf{p} \wedge \Diamond \mathsf{r}$. In the next definition we make this intuition formal and define the truth of **ATL**$^+$ path formulae on finite sequences of states.

Definition 5 ($\models^k$) *Let M be a* **CGS***, λ be path in M, and $k \in \mathbb{N}$. The semantics $\models^k$ is defined by the following clauses:*

- $M, \lambda \models^k \neg\gamma$ *iff* $M, \lambda \not\models^k \gamma$*;*
- $M, \lambda \models^k \gamma \wedge \delta$ *iff* $M, \lambda \models^k \gamma$ *and* $M, \lambda \models^k \delta$*;*
- $M, \lambda \models^k \bigcirc \varphi$ *iff* $M, \lambda[1] \models \varphi$ *and* $k > 1$*; and*
- $M, \lambda \models^k \varphi \mathcal{U} \psi$ *iff there is an* $i < k$ *such that* $M, \lambda[i] \models \psi$ *and* $M, \lambda[j] \models \varphi$ *for all* $0 \leq j < i$*;*

Essentially, we consider the first k states on a path in order to see whether a formula is made true on it.

Proposition 4 *Let M be a* **CGS***, $h \in St^+$, and γ a flat* **ATL**$^+$ *path formula. Moreover, let λ^h be any infinite extension of h. The problem whether $M, \lambda^h \models^{|h|} \gamma$ is in* **P** *wrt to the length of h and the length of the formula γ and can be done in time $\mathcal{O}(|h| \cdot |\gamma| + |\gamma|^2)$; and even in time $\mathcal{O}(|h| \cdot |\gamma|)$ if the formula is given in prefix form.*

Proof For each $\gamma' \in \mathcal{APF}(\gamma)$ we can check whether γ' holds on h in $\mathcal{O}(|h|)$ steps; accordingly, we replace γ' by $\top$ (resp. $\bot$) if $M, \lambda^h \models^{|h|} \gamma'$ (resp. $M, \lambda^h \not\models^{|h|} \gamma'$). Following this procedure for at most $|\gamma|$ times we end up with a propositional formula build from $\top$ and $\bot$. Now, we can easily check whether the

formula evaluates to true in $\mathcal{O}(|\gamma|^2)$ steps; and even in time $\mathcal{O}(|\gamma|)$ if the formula is given in prefix form. This can be achieved by simple rewrite rules. ■

In the following we define a strategy that ensures that a formula is true on the initial history of length k; thereafter, we show that the existence of such strategies ensures that there also is a "normal" strategy.

Definition 6 (k-witness strategy) *We say that a strategy s_A is a k-*witness strategy *for (M, q, γ) if for all $\lambda \in out(q, s_A)$ we have that $M, \lambda \models^k \gamma$.*

The following theorem is essential for our model checking algorithm. The result ensures that the existence of a winning strategy can be decided by only guessing the first k-steps of a k-witness strategy.

Theorem 5 *$M, q \models \langle\langle A\rangle\rangle\gamma$ iff there is a $|St_M| \cdot |\mathcal{APF}(\gamma)|$-witness strategy for (M, q, γ).*

Proof "⇒:" Let s_A be a strategy for (M, q, γ). Since $|wit_M(\lambda, \gamma)| \leq |\mathcal{APF}(\gamma)|$ for any path λ; by Lemma 3 there is a strategy s'_A for (M, q, γ) such that $\max wit_M(\lambda, \gamma) \leq |St_M| \cdot |\mathcal{APF}(\gamma)|$ for all $\lambda \in out(q, s_A)$. (A segment can have a length of at most $|St|$.) So, we have that s'_A is a $|St_M| \cdot |\mathcal{APF}(\gamma)|$-witness strategy for (M, q, γ).

"⇐": Now assume there is a $k := |St_M| \cdot |\mathcal{APF}(\gamma)|$-witness strategy s_A for (M, q, γ) and no strategy s'_A for (M, q, γ). Clearly, every witness position can also be made true by the "normal" strategy s'_A. The only reason for the non-existence of the latter can be that there are more witness positions for atomic formulas, namely for those false wrt the k-witness strategy s_A. That is, a formula γ is witnessed *after* k steps wrt s_A.

More formally, for any k-witness strategy s_A there must be a $\gamma' \in \mathcal{APF}(\gamma)$ such that for all paths $\lambda \in out(q, s_A)$ it holds that $witpos_M(\lambda, \gamma') > k$ or $witpos_M(\lambda, \gamma') = -1$ and there must exist at least one path λ' such that $witpos_M(\lambda', \gamma') > k$. Note, since such a property holds for any k-strategy the agents in A have no strategy to avoid a state witnessing γ'. Moreover, since $witpos_M(\lambda', \gamma') > k \geq |St_M|$ the opponents can also enforce another path λ'' such that $witpos_M(\lambda'', \gamma') \leq k$ following the same reasoning as applied in Lemma 3. This contradicts the existence of any k-witness strategy for (M, q, γ). ■

In the next theorem we construct an alternating Turing machine that solves the model checking problem.

Theorem 6 *Let $\varphi \equiv \langle\langle A\rangle\rangle\gamma$ be a flat $\mathbf{ATL}^+_{\mathbf{IR}}$ formula, M a concurrent game structure, and q a state in it. Then, there is an alternating Turing machine with $\mathcal{O}(nl)$ alternations that terminates in $\mathcal{O}(nkl)$ steps, and returns "yes" iff $M, q \models \varphi$, where l is the length of φ, k is the number of agents, and n the number of states in M.*

Proof The idea behind the algorithm can be summarized as follows: coalition A acts as a collective "verifier", and the rest of the agents plays the role of a collective "refuter" of the formula. We first transform γ to its negation normal form.[4] Next, we allow the verifier to nondeterministically construct A's strategy step by step for the first $|St_M| \cdot |\mathcal{APF}(\gamma)| \leq nl$ rounds (k steps each), while the refuter guesses the most damaging responses of $\mathbb{A}\mathrm{gt} \setminus A$. This is achieved by alternatingly guessing the best actions of A (Or-states) and the worst responses of the opponents (And-states); the number of alternations is bounded by $\mathcal{O}(nl)$ and the number of steps by $\mathcal{O}(nlk)$.

These guessed steps give us a finite path h (of length of at most nl) that is the outcome of the best strategy of A against the worst course of events. Then, we implement the game-theoretical semantics of propositional logic [8] as a game between the verifier (who controls disjunction) and the refuter (controlling conjunction) requiring at most $\mathcal{O}(l)$ steps. The game reduces the truth value of γ to a (possibly negated) atomic subformula γ_0. Finally, we check if $h' \models^{|St_M| \cdot |\mathcal{APF}(\gamma)|} \gamma_0$, and return the answer (h' denotes any infinite extension of h; it is not constructed/required by the machine). The latter can be done in deterministic polynomial time (cf. Proposition 4). The correctness of the construction follows from Theorem 5. It is now sufficient to observe that $\mathcal{O}(nlk + l + nl) = \mathcal{O}(nkl)$ ■

The following corollary is immediate from the fact that problems solvable in (non-deterministic) polynomial time by alternating Turing machines can also be solved by ordinary (non-alternating) Turing machines in polynomial space.

Corollary 7 *Model checking* $\mathbf{ATL^+_{IR}}$ *is* **PSPACE***-complete with respect to the number of states and agents in the model and the length of the formula. It is* **PSPACE***-complete even for turn-based models with two agents and "flat"* $\mathbf{ATL^+_{IR}}$ *formulae.*

3.3 Correcting Related Results: Alternating Transition Systems and Implicit CGS

Concurrent game structures specify transitions through a function that defines state transformations for *every combination of simultaneous actions* from $\mathbb{A}\mathrm{gt}$. In other words, transitions are given through an array that defines the outcome state for every combination of a state with k actions available at that state. This is clearly a disadvantage from the computational point of view, since the array is in general exponential with respect to the number of agents: more precisely, we have that $m = O(nd^k)$, where m is the number of (labeled) transitions in the model, n is the number of states, d is the maximal number of choices per state, and k is the number of agents.

[4] I.e., so that negation occurs only in front of atomic path subformulae.

Two variants of game structures overcome this problem. In *alternating transition systems* (**ATS**), used as models in the initial semantics of **ATL** [1, 2], agents' choices are state transformations themselves rather than abstract labels. In *implicit concurrent game structures* [9], the transition array is defined by use of Boolean expressions. **ATS** and implicit **CGS** do not hide exponential blowup in a parameter of the model checking problem (m), and hence the complexity of model checking for these representations is perhaps more meaningful than the results obtained for "standard" **CGS**. In [9], Laroussinie et al. claim that model checking **ATL**$^+$ against **ATS** as well as implicit **CGS** is $\mathbf{\Delta_3^P}$-complete. Since the proofs are actually based on the flawed result from [14], both claims are worth a closer look. We will briefly summarize both kinds of structures and give correct complexity results in this section.

Alternating Transition Systems. An **ATS** is a tuple $M = \langle \mathbb{A}\mathrm{gt}, St, \Pi, \pi, \delta \rangle$, where $\mathbb{A}\mathrm{gt}, St, \Pi, \pi$ are like in a **CGS**, and $\delta : St \times \mathbb{A}\mathrm{gt} \to 2^{2^{St}}$ is a function that maps each pair $(state, agent)$ to a non-empty family of choices with respect to possible next states. The idea is that, at state q, agent a chooses a set $Q_a \in \delta(q, a)$ thus forcing the outcome state to be from Q_a. The resulting transition leads to a state which is in the intersection of all Q_a for $a \in \mathbb{A}\mathrm{gt}$. Since the system is required to be deterministic (given the state and the agents' decisions), $Q_{a_1} \cap ... \cap Q_{a_k}$ must always be a singleton.

Implicit CGS. An implicit **CGS** is a concurrent game structure where, in each state q, the outgoing transitions are defined by a finite sequence $((\varphi_1, q_1), ..., (\varphi_n, q_n))$. In the sequence, every q_i is a state, and each φ_i is a Boolean combination of propositions $\hat{\alpha}^a$, where $\alpha \in d(a, q)$; $\hat{\alpha}^a$ stands for "agent a chooses action α". The transition function is now defined as: $o(q, \alpha_1, ..., \alpha_k) = q_i$ *iff* i *is the lowest index such that* $\{\hat{\alpha_1}^1, ..., \hat{\alpha_k}^k\} \models \varphi_i$. It is required that $\varphi_n \equiv \top$, so that no agent can enforce a deadlock.

Model Checking ATL$^+$ Is PSPACE-Complete Again. Contrary to [9, Section 3.4.1], where model checking **ATL**$^+$ with respect to both **ATS** and implicit **CGS** is claimed to be $\mathbf{\Delta_3^P}$-complete, we establish the complexity as **PSPACE**.

Theorem 8 *Model checking* $\mathbf{ATL^+_{IR}}$ *for* **ATS** *and implicit* **CGS** *is* **PSPACE***-complete with respect to the number of states, actions, and agents in the model, and the length of the formula. It is* **PSPACE***-complete even for turn-based models with two agents and "flat"* **ATL**$^+$ *formulae.*

Proof sketch Lower bound. We observe that the number of transitions in a turn-based **CGS** is linear in the number of states (n), agents (k), and actions (d). Moreover, each turn-based **CGS** has an isomorphic **ATS**, and an isomorphic implicit **CGS**; the transformation takes $O(nd)$ steps. This, together with the reduction from Section 3.1, gives us **PSPACE**-hardness wrt n, k, d

and the length of the formula (l) for model checking $\mathbf{ATL}^+$ against **ATS** as well as implicit **CGS**.

Upper bound. A close inspection of the proof of Theorem 6 reveals that it can be as well applied to **ATS** and implicit **CGS**. We also recall that the algorithm describes an alternating Turing Machine that runs in polynomial time. ■

4 Adding Fairness to ATL$^+$

Fairness conditions allow to focus on computations where no agent is neglected wrt given resources (e.g., access to power supply, processor time, etc.). Fairness is extremely important in an asynchronous composition of agents. In general, it may happen that requests of group $A \subsetneq \mathbb{A}\mathrm{gt}$ are postponed forever in favor of actions from other agents. In consequence, if we want to state any positive property about what A can achieve, we need to refer explicitly only to paths where A's actions are always eventually executed. To this end, it is enough to augment $\mathbf{ATL}^+$ with the "always eventually" combination $\Box\Diamond$ as an additional primitive.

4.1 EATL$^+$

$\mathbf{EATL}^+$ is a subset of $\mathbf{ATL}^*$ defined as follows:

$$\varphi ::= \mathsf{p} \mid \neg\varphi \mid \varphi \wedge \varphi \mid \langle\langle A\rangle\rangle\gamma, \text{where } \gamma ::= \neg\gamma \mid \gamma \wedge \gamma \mid \bigcirc\varphi \mid \varphi\,\mathcal{U}\,\varphi \mid \Box\Diamond\varphi.$$

We note that $\mathbf{EATL}^+$ is strictly more expressive than **ATL** (it follows from the fact that **ECTL** is strictly more expressive than **CTL** [6]), and hence also more expressive than $\mathbf{ATL}^+$ (clearly, Boolean combinations of path formulae cannot help to emulate the $\Box\Diamond$-modality). As before, we use the notation $\mathbf{EATL}^+_{\mathbf{IR}}$ to refer to the language of $\mathbf{EATL}^+$ used with the perfect recall semantics.

Moreover, in order to reason about the outcome of fair computations in model M, it suffices to do the following. First, we add to M special propositions act_i, one per agent i. The propositions indicate which agent has executed the most recent action. Now, for example, the $\mathbf{EATL}^+$ $\langle\langle 1,2\rangle\rangle\big((\bigwedge_i \Box\Diamond\mathsf{act}_i) \rightarrow \Diamond\mathsf{cleanRoom}\big)$ says that agents 1 and 2 can cooperate to make the room clean *for every course of events on which no agent is stalled forever*.

4.2 Model Checking EATL$^+$

In this section we extend the construction from Section 3.2 to obtain an algorithm for $\mathbf{EATL}^+_{\mathbf{IR}}$. Firstly, an $\mathbf{EATL}^+$-path formula γ is called $\Box\Diamond$-*atomic* if it has the form $\Box\Diamond\varphi_1$. For $\varphi \in \mathbf{EATL}^+$ we denote the set of all $\Box\Diamond$-atomic

flat path subformulae of φ by $\mathcal{APF}^\infty(\varphi)$. Secondly, we define the *witnesses for* $\gamma \equiv \Box\Diamond\varphi$. Here, it is a *set* $wit_M^\infty(\lambda,\gamma)$ of witnesses for a flat atomic formula; moreover, this set is either infinite or empty. If $\lambda \not\models \Box\Diamond\varphi$ then $wit_M^\infty(\lambda,\gamma) = \emptyset$; and if $\lambda \models \Box\Diamond\varphi$ then $wit_M^\infty(\lambda,\gamma) = \{i \mid M,\lambda[i] \models \varphi\}$.

In the following we generalize the definition of a segment.

Definition 7 (γ-segment, strict) *A* γ-segment *on a path* λ *is a tuple* $(i,j) \in \mathbb{N}_0^2$ *with* $i < j$ *such that* $\lambda[i] = \lambda[j]$ *and for each* $\gamma' \in \mathcal{APF}^\infty(\gamma)$ *with* $wit_M^\infty(\lambda,\gamma') \neq \emptyset$ *there is a witness* $w \in wit_M^\infty(\lambda,\gamma')$ *such that* $i \leq w \leq j$.

We call a γ*-segment* (i,j) strict *if there is no other* γ*-segment* (k,l) *with* $i \leq k \leq l \leq j$ *where* $i < k$ *or* $l < j$.

The next proposition shows that such γ-segments do always exist on paths on which some $\Box\Diamond$-atomic flat formula is true.

Proposition 9 *Let* s_A *be a strategy for* (M,q,γ). *Then, for all paths* $\lambda \in out(q,s_A)$ *and* $t \in \mathbb{N}$ *there is a strict* γ*-segment* (i,j) *on* λ *with* $i \geq t$.

Proof Suppose there is a path in the outcome that does not contain such a γ-segment. Then, as the set of states is finite there must be some position $l \geq t$ on λ such that $\lambda[l,\infty]$ does not contain a witness for some $\gamma' \in \mathcal{APF}^\infty(\gamma)$ with $wit_M(\lambda,\gamma') \neq \emptyset$. But this contradicts $wit_M(\lambda,\gamma') \neq \emptyset$. If there is no $\Box\Diamond$-formula true on a path the condition is trivially true. ∎

Lemma 10 *Let* $M,q \models \langle\langle A\rangle\rangle\gamma$. *Then, there is a strategy* s_A *for* (M,q,γ) *such that any strict* γ*-segment* (i,j) *that contains no more witnesses for any formula from* $\mathcal{APF}(\gamma)$ *contains at most* $|St_M| \cdot |\mathcal{APF}^\infty(\gamma)|$ *states.*

Proof We proceed similar to Lemma 3 to make all eventualities from $\mathcal{APF}(\gamma)$ true. Then, we modify the strategy to a strategy s'_A such that any segment (i_l,j_l) contained in any strict γ-segment (i,j) contains some witness of $wit_M^\infty(\lambda,\gamma')$ for each $\gamma' \in \mathcal{APF}^\infty(\gamma)$ for that the witness set it non-empty on λ (and which does not contain any more witnesses from formulae from $\mathcal{APF}(\gamma)$). Now, we consider any segment, say (i_l,j_l), contained in (i,j). If all formulae $\gamma' \in \mathcal{APF}^\infty(\gamma)$ with $wit_M^\infty(\lambda,\gamma') \neq \emptyset$ that have a witness in (i_l,j_l) do also have a witness inside (i,j) but outside (i_l,j_l) then we modify s'_A such that (i_l,j_l) is "removed" from the γ-segment (i,j) by applying the reduction of Lemma 3. The resulting γ-segment (i,j') is $j_l - i_l + 1$ states shorter than (i,j). Applied recursively, this procedure results in a γ-segment that contains at most $|\mathcal{APF}^\infty(\gamma)|$ necessary segments which are interconnected by a minimal number of states that do not contain unnecessary segments. The number of states of each segment plus the number of intermediate states between two segments is at most $|St_M|$. Hence, the γ-segment contains at most $|St_M|(|\mathcal{APF}^\infty(\gamma)|)$ states. ∎

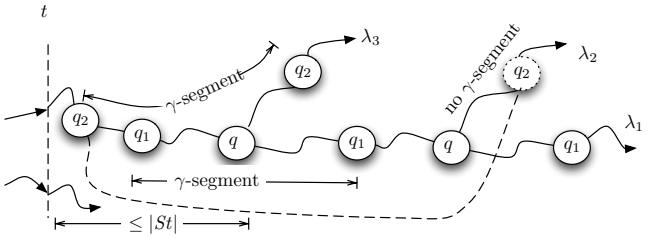

Figure 6: Proof idea of Theorem 11.

In the following we extend the finite path semantics such that it can deal with $\Box\Diamond$-atomic flat formulae.

Definition 8 ($\models^k$ for $\mathbf{EATL}^+$) *The semantics from Definition 5 is extended to* $\mathbf{EATL}^+$*-formulae by adding the following clause:* $M, \lambda \models^k \Box\Diamond\gamma$ *iff there is some* $i < k$ *such that* $M, \lambda[i, \infty] \models^k \gamma$*;*

The notion of a k-witness strategy is given as in Definition 6 by using the semantics just defined: s_A is a *k-witness* strategy for (M, q, γ) if for all $\lambda \in out(q, s_A)$ we have that $M, \lambda \models^k \gamma$.

The analog of Theorem 5 for $\mathbf{EATL}^+_{\mathbf{IR}}$ is given next.

Theorem 11 *We have that* $M, q \models \langle\langle A\rangle\rangle\gamma$ *iff there is a* $|St_M| \cdot (1 + |\mathcal{APF}(\gamma)| + |\mathcal{APF}^\infty(\gamma)|)$*-witness strategy for* (M, q, γ)*.*

Proof "$\Rightarrow$": Let s_A be a strategy for (M, q, γ). Then, we modify s_A according to Lemma 3 and obtain a strategy s'_A such that on all paths λ of the outcome of this new strategy and for all formula $\gamma' \in \mathcal{APF}(\gamma)$ with a witness on λ we have that $wit_M(\lambda, \gamma') \leq |St| \cdot |\mathcal{APF}(\gamma)| =: t$. We modify s'_A to a strategy s''_A according to Proposition 9 and Lemma 10. Finally, the states between t and the start of the strict γ-segment can be shrunk up to at most $|St|$-many states, by modifying the strategy again according to Lemma 3 (cf. Figure 6). The resulting strategy is a $|St_M| \cdot (1 + |\mathcal{APF}(\gamma)| + |\mathcal{APF}^\infty(\gamma)|)$-witness strategy for (M, q, γ).

"$\Leftarrow$": Now assume there is a $k := |St_M| \cdot (1 + |\mathcal{APF}(\gamma)| + |\mathcal{APF}^\infty(\gamma)|)$-witness strategy for (M, q, γ) and no strategy for (M, q, γ). If this is caused by a formula from $\mathcal{APF}(\gamma)$ the reasoning is as in the proof of Theorem 5. Now we consider the case if it is caused by a formula from $\mathcal{APF}^\infty(\gamma)$. Then, for any k-strategy s_A there must be a $\gamma' \in \mathcal{APF}^\infty(\gamma)$ such that for some paths $\lambda_1 \in out(q, s_A)$ it holds that $M, \lambda_1 \models^k \gamma'$ but $M, \lambda_2 \not\models \gamma'$ where λ_2 equals λ_1 up to position k. We show that this cannot be the case. $M, \lambda_1 \models^k \gamma'$ implies that γ' has a witness in the initial γ-segment on λ_1 (cf. the initial γ-segment on

λ_1 with start and end state q_1 in Figure 6). So, there must be a state q and an outgoing path λ_2 containing no more γ-segments. However, this state and the outgoing path must also be present in the initial γ-segment on the path λ_1. On λ_3, however, (see Fig. 6) there must also be an initial γ-segment. If it starts within q_1 and q on λ_3 it must also be present on λ_2. So, suppose the initial γ-segment on λ_3 with start and end state q_2 begins before q_1. This gives us a (non-strict) γ-segment on λ_2 (shown by the dotted line) and of course, this segment can also be reached on the outgoing path λ_2 going through state q on λ_1. Applying this reasoning recursively proves that each of these paths contains infinitely many γ-segments. This contradicts the assumption that $M, \lambda_2 \not\models \gamma'$. ■

The previous result allows to construct an alternating Turing machine with a fixed number of alternations to solve the model checking problem (cf. Theorem 6).

Theorem 12 *Let φ be a flat* $\mathbf{EATL}^+$ *formula, M be a concurrent game structure, and q a state in it. Then, there is an alternating Turing machine with $\mathcal{O}(nl)$ alternations that terminates in $\mathcal{O}(nkl)$ steps, and returns "yes" iff $M, q \models \varphi$ where l is the length of φ, k is the number of agents, and n the number of states in M.*

Proof sketch The proof is done analogously to the one of Theorem 6. Here one also has to incorporate the clause for $\Box\Diamond$-atomic formulae. The correctness follows from Theorem 11 and from the fact that $|St_M| \cdot (1 + |\mathcal{APF}(\varphi)| + |\mathcal{APF}(\varphi)|) \cdot |\mathbb{A}\mathrm{gt}| + |\gamma| + |St_M| \cdot |\gamma| \leq 3nkl = \mathcal{O}(nkl)$. ■

Finally, we get the following result as a combination of Theorem 12 and Theorem 2.

Corollary 13 *Model checking* $\mathbf{EATL}^+_{\mathbf{IR}}$ *is* **PSPACE***-complete with respect to the number of states and agents in the model and the length of the formula. It is* **PSPACE***-complete even for turn-based models with two agents and "flat"* $\mathbf{EATL}^+_{\mathbf{IR}}$ *formulae.*

5 Conclusions

In this paper, we corrected a result concerning the model checking complexity of $\mathbf{ATL}^+$ with respect to agents that remember the whole history of the game. In an otherwise excellent study [14], the problem was "proved" to be $\mathbf{\Delta}^{\mathbf{P}}_{\mathbf{3}}$-complete. Our amendment is rather pessimistic as we showed that the problem is in fact **PSPACE**-complete. In consequence, the results on model checking $\mathbf{ATL}^+_{\mathbf{IR}}$, reported in [9], are also incorrect. On the other hand, we also showed that adding fairness conditions does not increase the complexity further, which is definitely good news.

Although $\mathbf{ATL}^+$ is a fragment of $\mathbf{ATL}^*$ which has not yet attracted much attention, we believe that our results are significant for several reasons. First of all, we correct a widely believed "result" about model checking $\mathbf{ATL}^+$, and that is important on its own. Moreover, several other existing results concerning variants of the model checking problem have been based on the $\mathbf{\Delta_3^P}$-completeness for $\mathbf{ATL}^+_{\mathbf{IR}}$, and thus needed to be rectified as well. Finally, as we tried to argue, $\mathbf{ATL}^+$ allows to express interesting properties that cannot be expressed in $\mathbf{ATL}$ – or, if they can be expressed, $\mathbf{ATL}^+$ provides a more succinct and more intuitive presentation. That is especially true for its extended variant $\mathbf{EATL}^+$, and this makes the logic valuable for specification and verification of multi-agent systems if one wants to avoid the prohibitive complexity of model checking with full $\mathbf{ATL}^*$.

References

[1] R. Alur, T. A. Henzinger, and O. Kupferman. Alternating-time Temporal Logic. In *Proceedings of the 38th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 100–109. IEEE Computer Society Press, 1997.

[2] R. Alur, T. A. Henzinger, and O. Kupferman. Alternating-time Temporal Logic. *Lecture Notes in Computer Science*, 1536:23–60, 1998.

[3] R. Alur, T. A. Henzinger, and O. Kupferman. Alternating-time Temporal Logic. *Journal of the ACM*, 49:672–713, 2002.

[4] N. Bulling and W. Jamroga. Agents, beliefs and plausible behaviour in a temporal setting. In *Proceedings of AAMAS'07*, pages 570–577, 2007.

[5] E. Clarke and E. Emerson. Design and synthesis of synchronization skeletons using branching time temporal logic. In *Proceedings of Logics of Programs Workshop*, volume 131 of *Lecture Notes in Computer Science*, pages 52–71, 1981.

[6] E. A. Emerson. Temporal and modal logic. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, pages 995–1072. Elsevier Science Publishers, 1990.

[7] A. Harding, M. Ryan, and P.-Y. Schobbens. Approximating atl* in atl. In *VMCAI '02: Revised Papers from the Third International Workshop on Verification, Model Checking, and Abstract Interpretation*, pages 289–301, London, UK, 2002. Springer-Verlag.

[8] J. Hintikka. *Logic, Language Games and Information*. Clarendon Press : Oxford, 1973.

[9] F. Laroussinie, N. Markey, and G. Oreiby. On the expressiveness and complexity of atl. *LMCS*, 4:7, 2008.

[10] F. Laroussinie, N. Markey, and P. Schnoebelen. Model checking CTL+ and FCTL is hard. In *Proceedings of FoSSaCS'01*, volume 2030 of *Lecture Notes in Computer Science*, pages 318–331. Springer, 2001.

[11] A. Lomuscio and M. Sergot. Deontic interpreted systems. *Studia Logica*, 75(1):63–92, 2003.

[12] A. Lomuscio and M. Sergot. A formalization of violation, error recovery, and enforcement in the bit transmission problem. *Journal of Applied Logic*, 1, 2003. To appear. Preliminary version published in the Proceedings of DEON'02.

[13] C. Papadimitriou. *Computational Complexity*. Addison Wesley : Reading, 1994.

[14] P. Y. Schobbens. Alternating-time logic with imperfect recall. *Electronic Notes in Theoretical Computer Science*, 85(2), 2004.

[15] T. Wilke. CTL+ is exponentially more succint than CTL. In *Proceedings of FST&TCS '99*, Lecture Notes in Computer Science. Springer Verlag, 1999.

[16] B. Wozna and A. Lomuscio. A logic for knowledge, correctness, and real time. In *Proceedings of CLIMA V*, Lecture Notes in Computer Science. Springer Verlag, 2004.